EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	222	(operand same op$code near4 fetch$3)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:34
L2	137	(operand same op$code near4 fetch$3) same (cache or memory)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:37
L3	22	(operand same op$code near4 fetch$3) with (cache or memory)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:41
L4	6	(operand same op$code near4 fetch$3) same (cache or memory) same (emulat$4 or simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:38
L5	9	((operand adj (cache or memory)) and (op$code adj (cache or memory)))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:08
L6	0	operand adj2 cache and op$code adj2 cache near4 fetch$3	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:51
L7	2	operand adj2 cache and op$code adj2 cache and fetch$3	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:52
L8	26	"717".clas. and (operand same op$code near4 (fetch$3 or receiv$3))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:54
L9	25	l8 and @ad<"20031205"	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 10:54

EAST Search History

L10	0	"717".clas. and ((operand adj (cache or memory)) and (op$code adj (cache or memory)))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:31
L11	440	((717/118) or (717/138) or (717/148)).CCLS.	USPAT; USOCR	OR	OFF	2007/03/28 11:21
L12	6	l11 and (op$code same operand near4 (fetch$3 or receiv$3))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:24
L13	519	(emulat$4 or simulat$4) same (processor and (external near3 memory))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:26
L14	28	((operand adj3 cache) and (op$code adj3 cache))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:27
L15	0	l13 and l14	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:27
L16	2357	operand with op$code	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:27
L17	27	l13 and l16	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:28
L18	0	"703".clas. and ((operand adj (cache or memory)) and (op$code adj (cache or memory)))	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:31
L19	5	sollom.inv.	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:33

EAST Search History

L20	548043	williams.inv.	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:34
L21	3	l20 and (operand same op$code near4 fetch$3)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:35
L22	1712	l20 and (operand or op$code)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:35
L23	301	l20 and (operand and op$code)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:35
L24	20	l20 and (operand and op$code same emulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:40
L25	79	(independent or separat$3) with (op$code and operand)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:41
L26	11	(independent or separat$3) near2 (op$code and operand)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:51
L27	565	compar$4 same (op$code and operand)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:51
L28	8	compar$4 same (op$code and operand) same (emulat$4 or simulat$5)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:53
L30	154	compar$4 with (op$code and operand)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:54

EAST Search History

L31	0	compar$4 with (op$code and operand) with (emulat$4 or simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:54
L32	46	compar$4 with (op$code and operand) and (emulat$4 or simulat$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; IBM_TDB	OR	ON	2007/03/28 11:54

Terms used **comparing opcode operand emulation** Found **162** of **198,991**

Sort results by relevance Save results to a Binder Try an Advanced Search

Display results expanded form Search Tips Try this search in The ACM Guide

Open results in a new window

Results 1 - 20 of 162 Result page: **1** 2 3 4 5 6 7 8 9 next

Relevance scale

1 SSIM: a software levelized compiled-code simulator

L.-T. Wang, N. E. Hoover, E. H. Porter, J. J. Zasio

October 1987 **Proceedings of the 24th ACM/IEEE conference on Design automation DAC '87**

Publisher: ACM Press

Full text available: pdf(765.27 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents a new logic simulation technique that uses software levelized compiled-code (LCC) for synchronous designs. Three approaches are proposed: C source code, target machine code and interpreted code. The evaluation speed for the software LCC simulator (SSIM) is about 140,000 (gate) evaluations per second using C source code or target machine code, or 50,000 evaluations per second using interpreted code. It is about 40 to 100 times slower than the AIDA hardware LCC simulator, ...

2 A flexible VLSI core for an adaptable architecture

H. Mulder, P. Stravers

August 1989 **ACM SIGMICRO Newsletter , Proceedings of the 22nd annual workshop on Microprogramming and microarchitecture MICRO 22**, Volume 20 Issue 3

Publisher: ACM Press

Full text available: pdf(946.24 KB) Additional Information: full citation, abstract, references, citings, index terms

Two major limitations concerning the design of cost-effective application-specific architectures are the recurrent costs of system-software development and hardware implementation, in particular VLSI implementation, for each architecture. The SCalable ARChitecture Experiment (SCARCE) aims to provide a framework for application-specific processor design. The framework allows scaling of functionality, implementation complexity, and performance. The SCARCE framework consists and wil ...

3 Architectural issues: Modifications to the VAX-11/780 microarchitecture to support IEEE floating point arithmetic

David B. Aspinwall, Yale N. Patt

December 1983 **ACM SIGMICRO Newsletter**, Volume 14 Issue 4

Publisher: ACM Press

Full text available: pdf(594.85 KB) Additional Information: full citation, abstract, references

The VAX-11/780 was designed specifically to implement the VAX architecture. As such, it does not support the IEEE standard for floating point arithmetic. A project was undertaken

to provide this support by modifying the 11/780 microarchitecture. Our objective was to produce a microengine that would efficiently execute a modified VAX instruction set, in particular, one that executes VAX floating point instructions consistent with the IEEE standard. We made minimal changes to the 11/780 hardware, ...

4 Randomized instruction set emulation

Elena Gabriela Barrantes, David H. Ackley, Stephanie Forrest, Darko Stefanović
February 2005 **ACM Transactions on Information and System Security (TISSEC)**, Volume 8 Issue 1
Publisher: ACM Press

Full text available: pdf(374.44 KB) Additional Information: full citation, abstract, references, citings, index terms

Injecting binary code into a running program is a common form of attack. Most defenses employ a "guard the doors" approach, blocking known mechanisms of code injection. *Randomized instruction set emulation* (RISE) is a complementary method of defense, one that performs a hidden randomization of an application's machine code. If foreign binary code is injected into a program running under RISE, it will not be executable because it will not know the proper randomization. The pape ...

Keywords: Automated diversity, randomized instruction sets, software diversity

5 Generation of fast interpreters for Huffman compressed bytecode

Mario Latendresse, Marc Feeley
June 2003 **Proceedings of the 2003 workshop on Interpreters, virtual machines and emulators IVME '03**
Publisher: ACM Press

Full text available: pdf(323.22 KB) Additional Information: full citation, abstract, references, citings, index terms

Embedded systems often have severe memory constraints requiring careful encoding of programs. For example, smart cards have on the order of 1K of RAM, 16K of non-volatile memory, and 24K of ROM. A virtual machine can be an effective approach to obtain compact programs but instructions are commonly encoded using one byte for the opcode and multiple bytes for the operands, which can be wasteful and thus limit the size of programs runnable on embedded systems. Our approach uses canonical Huffman co ...

Keywords: Java, canonical Huffman code, code compression, decoder

6 The role of emulation in performance measurement and evaluation

Liba Svobodova, Roy Mattson
March 1976 **Proceedings of the 1976 ACM SIGMETRICS conference on Computer performance modeling measurement and evaluation SIGMETRICS '76**
Publisher: ACM Press

Full text available: pdf(840.52 KB) Additional Information: full citation, abstract, references, citings, index terms

Emulation of systems makes it possible to combine the predictive power of simulation with the advantages of measurement carried under a real system workload. An emulator is a microprogrammed implementation of the basic hardware machine. It can be easily instrumented to collect performance statistics on the instruction set processor (ISP) level and support performance measurement of different configurations and software of the emulated system. This paper describes the monitoring capabilities ...

7 Efficient macro-code emulation in hardwired pipelined processors

J. M. Mulder, R. J. Portier, A. Srivastava, R. in't Velt
January 1988 **Proceedings of the 21st annual workshop on Microprogramming and microarchitecture MICRO 21**
Publisher: IEEE Computer Society Press

Full text available: pdf(731.15 KB) Additional Information: full citation, abstract, references, citings, index terms

Traditionally microcoded computers have been the ideal machines for implementing scalable architectures. These machines easily implement application-specific functionality in microcode and they allow architecturally transparent variation of cost/performance by trading off application code, microcode, and hardware. In contrast, hardwired machines are intrinsically incapable of implementing scalability, because they only implement a single level of interpretation. Recent RISC designs have int ...

8 MICROSIM: A microinstruction simulator for teaching microprogramming and emulation
Francis P. Mathur
September 1977 **ACM SIGMICRO Newsletter , Proceedings of the 10th annual workshop on Microprogramming MICRO 10**, Volume 8 Issue 3
Publisher: IEEE Press, ACM Press

Full text available: pdf(753.43 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes in the format of a user's manual, along with an example, a program for teaching microprogramming and emulation. This program simulates at the microinstruction set level the architecture of a 16-bit microprogram-controlled minicomputer. This simulator, as an instructional vehicle, has been successfully used in a software engineering laboratory adjunct to a regular course in microprogramming and computer architecture.

9 Software and languages: Proteus: virtualization for diversified tamper-resistance
Bertrand Anckaert, Mariusz Jakubowski, Ramarathnam Venkatesan
October 2006 **Proceedings of the ACM workshop on Digital rights management DRM '06**
Publisher: ACM Press

Full text available: pdf(299.79 KB) Additional Information: full citation, abstract, references, index terms

Despite huge efforts by software providers, software protection mechanisms are still broken on a regular basis. Due to the current distribution model, an attack against one copy of the software can be reused against any copy of the software. Diversity is an important tool to overcome this problem. It allows for renewable defenses in space, by giving every user a different copy, and renewable defenses in time when combined with tailored updates. This paper studies the possibilities and limitation ...

Keywords: copyright protection, diversity, intellectual property, obfuscation, tamper-resistance, virtualization

10 An architecture framework for application-specific and scalable architectures
J. M. Mulder, R. J. Portier, A. Srivastava, R. in't Velt
April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the 16th annual international symposium on Computer architecture ISCA '89**, Volume 17 Issue 3
Publisher: ACM Press

Full text available: pdf(796.55 KB) Additional Information: full citation, abstract, references, citings, index terms

Two major limitations concerning the design of cost-effective application-specific

architectures are the recurrent costs of system-software development and hardware implementation, in particular VLSI implementation, for each architecture. The SCalable ARChitecture Experiment (SCARCE) aims to provide a framework for application-specific processor design. The framework allows scaling of functionality, implementation complexity, and performance. The SCARCE framework consists and wil ...

11 Automatic Formal Verification of Fused-Multiply-Add FPUs

Christian Jacobi, Kai Weber, Viresh Paruthi, Jason Baumgartner

March 2005 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 2 DATE '05**

Publisher: IEEE Computer Society

Full text available: pdf(241.82 KB) Additional Information: full citation, abstract, citings, index terms

In this paper we describe a fully-automated methodology for formal verification of fused-multiply-add floating point units (FPUs). Our methodology verifies an implementation FPU against a simple reference model derived from the processor's architectural specification, which may include all aspects of the IEEE specification including denormal operands and exceptions. Our strategy uses a combination of BDD- and SAT-based symbolic simulation. To make this verification task tractable, we use a combi ...

12 Functional verification of the equator MAP1000 microprocessor

Jian Shen, Jacob Abraham, Dave Baker, Tony Hurson, Martin Kinkade, Gregorio Gervasio, Chen-chau Chu, Guanghui Hu

June 1999 **Proceedings of the 36th ACM/IEEE conference on Design automation DAC '99**

Publisher: ACM Press

Full text available: pdf(73.73 KB) Additional Information: full citation, references, citings, index terms

13 MASCO: An academic exercise in computer design using microprogramming

Jack N. Fenner, Jeffery A. Schmidt, Houssam A. Halabi, Dharma P. Agrawal

December 1984 **ACM SIGMICRO Newsletter , Proceedings of the 17th annual workshop on Microprogramming MICRO 17**, Volume 15 Issue 4

Publisher: IEEE Press, ACM Press

Full text available: pdf(750.42 KB) Additional Information: full citation, abstract, references, index terms

The design of a host machine architecture for implementing an existing image machine architecture is considered. The proposed microarchitecture uses a microprogrammed control unit to emulate the Motorola MC6809 microprocessor architecture. The detailed hardware characteristics of the proposed machine are described and the microinstruction set is defined. The microroutines that emulate the MC6809 instruction set have been developed. A practical implementation using current technol ...

14 Design problems in emulating the MIX computer on the Microdata 1600

T. Don Dennis, O. G. Johnson

September 1976 **Proceedings of the 9th annual workshop on Microprogramming MICRO 9**

Publisher: ACM Press

Full text available: pdf(388.01 KB) Additional Information: full citation, abstract, references, index terms

This paper presents an overview of an emulator for the MIX computer written in Microdata 1600 microcode. The MIX computer thus emulated is a variant of the original MIX computer as described in Volume 1 of The Art of Computer Programming by Donald Knuth. Basic changes involve the utilization of 8 bit bytes along with the ASCII character code.

15 LimitLESS directories: A scalable cache coherence scheme

David Chaiken, John Kubiatowicz, Anant Agarwal

April 1991 **ACM SIGPLAN Notices , ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , Proceedings of the fourth international conference on Architectural support for programming languages and operating systems ASPLOS-IV**, Volume 26 , 19 , 25 Issue 4 , 2 , Special Issue

Publisher: ACM Press

Full text available: pdf(1.20 MB) Additional Information: full citation, references, citings, index terms

16 Languages: High performance annotation-aware JVM for Java cards

Ana Azevedo, Arun Kejariwal, Alex Veidenbaum, Alexandru Nicolau

September 2005 **Proceedings of the 5th ACM international conference on Embedded software EMSOFT '05**

Publisher: ACM Press

Full text available: pdf(158.29 KB) Additional Information: full citation, abstract, references, index terms

Early applications of smart cards have focused in the area of personal security. Recently, there has been an increasing demand for networked, multi-application cards. In this new scenario, enhanced application-specific on-card Java applets and complex cryptographic services are executed through the smart card Java Virtual Machine (JVM). In order to support such computation-intensive applications, contemporary smart cards are designed with built-in microprocessors and memory. As smart cards are h ...

Keywords: Java card, high performance, superoperators, virtual machine

17 Improving Java performance using hardware translation

Ramesh Radhakrishnan, Ravi Bhargava, Lizy K. John

June 2001 **Proceedings of the 15th international conference on Supercomputing ICS '01**

Publisher: ACM Press

Full text available: pdf(254.91 KB) Additional Information: full citation, abstract, references, citings, index terms

State of the art Java Virtual Machines with Just-In-Time (JIT) compilers make use of advanced compiler techniques, run-time profiling and adaptive compilation to improve performance. However, these techniques for alleviating performance bottlenecks are more effective in long running workloads, such as server applications. Short running Java programs, or client workloads, spend a large fraction of their execution time in compilation instead of useful execution when run using JIT compilers. In ...

18 Techniques for obtaining high performance in Java programs

Iffat H. Kazi, Howard H. Chen, Berdenia Stanley, David J. Lilja

September 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 3

Publisher: ACM Press

Full text available: pdf(816.13 KB) Additional Information: full citation, abstract, references, citings, index terms

This survey describes research directions in techniques to improve the performance of programs written in the Java programming language. The standard technique for Java execution is interpretation, which provides for extensive portability of programs. A Java interpreter dynamically executes Java bytecodes, which comprise the instruction set of the Java Virtual Machine (JVM). Execution time performance of Java programs can be improved through compilation, possibly at the expense of portabili ...

Keywords: Java, Java virtual machine, bytecode-to-source translators, direct compilers, dynamic compilation, interpreters, just-in-time compilers

19 Architecture: The microarchitecture of FPGA-based soft processors

Peter Yiannacouras, Jonathan Rose, J. Gregory Steffan
September 2005 **Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems CASES '05**
Publisher: ACM Press

Full text available: pdf(202.27 KB) Additional Information: full citation, abstract, references, citings, index terms

As more embedded systems are built using FPGA platforms, there is an increasing need to support processors in FPGAs. One option is the *soft processor*, a programmable instruction processor implemented in the reconfigurable logic of the FPGA. Commercial soft processors have been widely deployed, and hence we are motivated to understand their microarchitecture. We must re-evaluate microarchiteture in the soft processor context because an FPGA platform is significantly different than an ASIC ...

Keywords: ASIP, FPGA, Nios, RTL generation, SPREE, application specic tradeoff, embedded processor, exploration, microarchitecture, pipeline, soft processor

20 Empirical analysis of the mesa instruction set

Richard E. Sweet, James G. Sandman
March 1982 **ACM SIGPLAN Notices , ACM SIGARCH Computer Architecture News , Proceedings of the first international symposium on Architectural support for programming languages and operating systems ASPLOS-I**, Volume 17 , 10 Issue 4 , 2
Publisher: ACM Press

Full text available: pdf(804.42 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes recent work to refine the instruction set of the Mesa processor. Mesa [8] is a high level systems implementation language developed at Xerox PARC during the middle 1970's. Typical systems written in Mesa are large collections of programs running on single-user machines. For this reason, a major design goal of the project has been to generate compact object programs. The computers that execute Mesa programs are implementations of a stack architecture [5]. The ...

Welcome United States Patent and Trademark Office

☐ Search Session History

BROWSE SEARCH IEEE XPLORE GUIDE

Edit an existing query or compose a new query in the Search Query Display.

Select a search number (#) to:

- Add a query to the Search Query Display
- Combine search queries using AND, OR, or NOT
- Delete a search
- Run a search

Wed, 28 Mar 2007, 12:26:38 PM EST

Search Query Display

Recent Search Queries

#1	((opcode interface and operand interface)<in>metadata)
#2	((opcode and operand and comparison and emulation) <in>metadata)
#3	((comparing opcode and operand)<in>metadata)
#4	((emulation<in>metadata) <and> (opcode<in>metadata)) <and> (operand<in>metadata)
#5	((emulation<in>metadata) <and> (opcode<in>metadata)) <and> (operand<in>metadata)
#6	((emulation or simulation<in>metadata) <and> (op-code<in>metadata))<and> (operand<in>metadata)

Indexed by Inspec®